



博客 (<http://blog.csdn.net/ref=toolbar>)

学院 (<http://edu.csdn.net/ref=toolbar>)

下载 (<http://download.csdn.net/ref=toolbar>)

GitChat (<http://gitbook.cn/?ref=csdn>)

更多 ▾



102



登录 (<https://passport.csdn.net/account/login?ref=toolbar>)

注册 (<https://passport.csdn.net/account/mobileRegister?ref=toolbar&action=mobileRegister>)

Spring Boot 入门



原创 2015年12月27日 15:41:29

标签: [spring](http://so.csdn.net/so/search/s.do?q=spring&t=blog) /

[springboot](http://so.csdn.net/so/search/s.do?q=springboot&t=blog) /



[boot](http://so.csdn.net/so/search/s.do?q=boot&t=blog)

205271



Spring Boot 入门

Spring Boot是Spring社区较新的一个项目。该项目的目的是帮助开发者更容易的创建基于Spring的应用程序和服务,让更多的人更快的对Spring进行入门体验,让Java开发也能够实现Ruby on Rails那样的生产效率。为Spring生态系统提供了一种固定的、约定优于配置风格的框架。

Spring Boot具有如下特性:

- 为基于Spring的开发提供更快的入门体验
- 开箱即用,没有代码生成,也无需XML配置。同时也可以修改默认值来满足特定的需求。
- 提供了一些大型项目中常见的非功能性特性,如嵌入式服务器、安全、指标、健康检测、外部配置等。
- Spring Boot并不是不对Spring功能上的增强,而是提供了一种快速使用Spring的方式。

Spring Boot 系列

1. Spring Boot 入门 (<http://blog.csdn.net/isea533/article/details/50278205>)
2. Spring Boot 属性配置和使用 (<http://blog.csdn.net/isea533/article/details/50281151>)
3. Spring Boot 集成MyBatis (<http://blog.csdn.net/isea533/article/details/50359390>)
4. Spring Boot 静态资源处理 (<http://blog.csdn.net/isea533/article/details/50412212>)
5. Spring Boot - 配置排序依赖技巧 (<http://blog.csdn.net/isea533/article/details/53975720>)
6. Spring Boot - DevTools 介绍 (<http://blog.csdn.net/isea533/article/details/70495714>)



isea533 (<http://blog.csdn.net/isea533>)

+ 关注

(<http://blog.csdn.net/isea533>)

码云

原创 225 粉丝 2878 喜欢 1018 41 (<https://github.com/isea533>)

他的最新文章

更多文章 (<http://blog.csdn.net/isea533>)

[数据][json格式] 2016年统计用区划代码和城乡划分代码 (<http://blog.csdn.net/isea533/article/details/78862295>)

配置 IDEA 启动的 JDK (<http://blog.csdn.net/isea533/article/details/78621930>)

MyBatis 通用 Mapper 实现原理 (<http://blog.csdn.net/isea533/article/details/78493852>)

Java TCP 抓包简单示例 (<http://blog.csdn.net/isea533/article/details/78450264>)

Maven 最佳实践之 · 一个好的 parent 依赖基础 (<http://blog.csdn.net/isea533/article/details/78449270>)

泰国语言学习 基础泰语

韩语入门 自学韩语

博主专栏

本文根据官方文档深入讲解一段代码

简单例子

Spring Boot建议使用Maven或Gradle，本文以Maven为例。



102 首先创建一个一般的Maven项目，有一个pom.xml和基本的 src/main/java 结构。



在 **pom.xml** 中写上如下内容：



```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0"
3     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4     xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xs
5 d/maven-4.0.0.xsd">
6     <modelVersion>4.0.0</modelVersion>
7
8     <groupId>com.github.abel533</groupId>
9     <artifactId>spring-boot</artifactId>
10    <version>1.0-SNAPSHOT</version>
11
12    <parent>
13        <groupId>org.springframework.boot</groupId>
14        <artifactId>spring-boot-starter-parent</artifactId>
15        <version>1.3.0.RELEASE</version>
16    </parent>
17
18    <dependencies>
19        <dependency>
20            <groupId>org.springframework.boot</groupId>
21            <artifactId>spring-boot-starter-web</artifactId>
22        </dependency>
23    </dependencies>
24
25    <build>
26        <plugins>
27            <plugin>
28                <groupId>org.springframework.boot</groupId>
29                <artifactId>spring-boot-maven-plugin</artifactId>
30                <dependencies>
31                    <dependency>
32                        <groupId>org.springframework</groupId>
33                        <artifactId>springloaded</artifactId>
34                        <version>1.2.5.RELEASE</version>
35                    </dependency>
36                </dependencies>
37            </plugin>
38        </plugins>
39    </build>
</project>
```

首先是增加了 **<parent>**

增加父pom比较简单，而且 spring-boot-starter-parent 包含了大量配置好的依赖管理，在自己项目添加这些依赖的时候不需要写 <version> 版本号。



Mybatis示例
(<http://blog.csdn.net/column/details/sample.html>)

(<http://blog.csdn.net/column/details/mybatis.html>)



Mybatis问题集
(<http://blog.csdn.net/column/details/mybatis.html>)

(<http://blog.csdn.net/column/details/mybatis.html>)



Spring Boot 入门
(<http://blog.csdn.net/column/details/spring-boot.html>)

(<http://blog.csdn.net/column/details/spring-boot.html>)

他的热门文章

Spring Boot 集成MyBatis (<http://blog.csdn.net/isea533/article/details/50359390>)
271714

Spring Boot 属性配置和使用 (<http://blog.csdn.net/isea533/article/details/50281151>)
213726

MyBatis Generator 详解 (<http://blog.csdn.net/isea533/article/details/42102297>)
206658

Spring Boot 入门 (<http://blog.csdn.net/isea533/article/details/50278205>)
204926

Mybatis极其(最)简(好)单(用)的一个分页插件 (<http://blog.csdn.net/isea533/article/details/23831273>)
173757



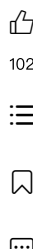
移民澳大利亚



it培训机构排名

雨水收集系统 迪拜十星级酒店
艺考文化课冲刺 希腊购房 世界十大..
住人集装箱 听英语学英语 插花入门
如何炒外汇入门 法语入门 小项目外包

使用父pom虽然简单，但是有些情况我们已经有父pom，不能直接增加 <parent> 时，可以通过如下方式：



```
1  <dependencyManagement>
2      <dependencies>
3          <dependency>
4              <!-- Import dependency management from Spring Boot -->
5              <groupId>org.springframework.boot</groupId>
6              <artifactId>spring-boot-dependencies</artifactId>
7              <version>1.2.3.RELEASE</version>
8              <type>pom</type>
9              <scope>import</scope>
10         </dependency>
11     </dependencies>
12 </dependencyManagement>
```

java.version属性

上面pom.xml虽然没有出现这个属性，这里要特别提醒。

Spring默认使用jdk1.6，如果你想使用jdk1.8，你需要在 pom.xml 的属性里面添加 java.version ，如下：

```
1  <properties>
2      <java.version>1.8</java.version>
3  </properties>
```

添加spring-boot-starter-web依赖

Spring通过添加 spring-boot-starter-* 这样的依赖就能支持具体的某个功能。

我们这个示例最终是要实现web功能，所以添加的是这个依赖。

更完整的功能列表可以查看：using-boot-starter-poms (<http://docs.spring.io/spring-boot/docs/1.2.3.RELEASE/reference/html/using-boot-build-systems.html#using-boot-starter-poms>)

添加spring-boot-maven-plugin插件

该插件支持多种功能，常用的有两种，第一种是打包项目为可执行的jar包。

在项目根目录下执行 `mvn package` 将会生成一个可执行的jar包，jar包中包含了所有依赖的jar包，只需要这一个jar包就可以运行程序，使用起来很方便。该命令执行后还会保留一个 `xxx.jar.original` 的jar包，包含了项目中单独的部分。

生成这个可执行的jar包后，在命令行执行 `java -jar xxxx.jar` 即可启动项目。

另外一个命令就是 `mvn spring-boot:run`，可以直接使用 tomcat （默认）启动项目。

在我们开发过程中，我们需要经常修改，为了避免重复启动项目，我们可以启用热部署。

Spring-Loaded 项目提供了强大的热部署功能，添加/删除/修改 方法/字段/接口/枚举 等代码的时候都可以热部署，速度很快，很方便。

想在 Spring Boot 中使用该功能非常简单，就是在 spring-boot-maven-plugin 插件下面添加依赖：

```
1 <dependencies>
2   <dependency>
3     <groupId>org.springframework</groupId>
4     <artifactId>springloaded</artifactId>
5     <version>1.2.5.RELEASE</version>
6   </dependency>
7 </dependencies>
```



¹⁰²添加以后，通过 `mvn spring-boot:run` 启动就支持热部署了。



注意：使用热部署的时候，需要IDE编译类后才能生效，你可以打开自动编译功能，这样在你保存修改的时候，类就自动重新加载了。



创建一个应用类

我们创建一个 `Application` 类：

```
1 @RestController
2 @EnableAutoConfiguration
3 public class Application {
4
5     @RequestMapping("/")
6     String home() {
7         return "Hello World!";
8     }
9
10    @RequestMapping("/now")
11    String hehe() {
12        return "现在时间: " + (new Date()).toLocaleString();
13    }
14
15    public static void main(String[] args) {
16        SpringApplication.run(Example.class, args);
17    }
18
19 }
```

注意

Spring Boot建议将我们 `main` 方法所在的这个主要的配置类配置在根包名下。

类似如下结构：



```
1  com
2  +- example
3      +- myproject
4          +- Application.java
5          |
6          +- domain
7              +- Customer.java
8              +- CustomerRepository.java
9              |
10             +- service
11                 +- CustomerService.java
12                 |
13             +- web
14                 +- CustomerController.java
```

在 `Application.java` 中有 `main` 方法。

因为默认和包有关的注解，默认包名都是当前类所在的包，例如 `@ComponentScan`，`@EntityScan`，`@SpringBootApplication` 注解。

@RestController

因为我们例子是写一个web应用，因此写的这个注解，这个注解相当于同时添加 `@Controller` 和 `@ResponseBody` 注解。

@EnableAutoConfiguration

Spring Boot建议只有一个带有该注解的类。

@EnableAutoConfiguration 作用：Spring Boot会自动根据你jar包的依赖来自动配置项目。例如当你项目下面有 `HSQLDB` 的依赖时，Spring Boot会创建默认的内存数据库的数据源 `DataSource`，如果你自己创建了 `DataSource`，Spring Boot就不会创建默认的 `DataSource`。

如果你不想让Spring Boot自动创建，你可以配置注解的 `exclude` 属性，例如：

```
1  @Configuration
2  @EnableAutoConfiguration(exclude={DataSourceAutoConfiguration.class})
3  public class MyConfiguration {
4  }
```

@SpringBootApplication

由于大量项目都会在主要的配置类上添加 `@Configuration`，`@EnableAutoConfiguration`，`@ComponentScan` 三个注解。

因此Spring Boot提供了 `@SpringBootApplication` 注解，该注解可以替代上面三个注解（使用Spring注解继承实现）。

home等方法



102



```
1  @RequestMapping("/")
2  String home() {
3      return "Hello World!";
4  }
5
6  @RequestMapping("/now")
7  String hehe() {
8      return "现在时间: " + (new Date()).toLocaleString();
9  }
```

这些方法都添加了 `@RequestMapping("xxx")`，这个注解起到路由的作用。

启动项目 `SpringApplication.run`

启动Spring Boot项目最简单的方法就是执行下面的方法：

```
1  SpringApplication.run(Application.class, args);
```

该方法返回一个 `ApplicationContext` 对象，使用注解的时候返回的具体类型

是 `AnnotationConfigApplicationContext` 或 `AnnotationConfigEmbeddedWebApplicationContext`，当支持web的时候是第二个。

除了上面这种方法外，还可以用下面的方法：

```
1  SpringApplication application = new SpringApplication(Application.class);
2  application.run(args);
```

`SpringApplication` 包含了一些其他可以配置的方法，如果你想做一些配置，可以用这种方式。

除了上面这种直接的方法外，还可以使用 `SpringApplicationBuilder`：

```
1  new SpringApplicationBuilder()
2      .showBanner(false)
3      .sources(Application.class)
4      .run(args);
```

当使用SpringMVC的时候由于需要使用子容器，就需要用到 `SpringApplicationBuilder`，该类有一个 `child(xxx...)` 方法可以添加子容器。

运行

在IDE中直接执行main方法，然后访问 `http://localhost:8080` 即可。

另外还可以用上面提到的 `mvn`，可以打包为可执行jar包，然后执行 `java -jar xxx.jar`。

或者执行 `mvn spring-boot:run` 运行项目。

项目启动后输出如下日志：



```
1 [INFO] Attaching agents: [F:\.m2\repository\org\springframework\springloaded\1.2.5.RELEASE
2 \springloaded-1.2.5.RELEASE.jar]
3
4 . ____ _ _ _ _
5 /\ / ____'_ _ _ _ _ _ _ _ _ _ \ \ \ \
6 ( ( )\___| '_ | '_ | | '_ \ \ \ \
7 \\/ ___| |_) | | | | | | (| | ) ) )
8 ' |___| .__| | | | | |___| / / / /
9 =====|_|=====|___/_/_/_/_/
10 :: Spring Boot :: (v1.2.3.RELEASE)
11
12 2015-12-12 22:26:35.298 INFO 9844 --- [ main] c.github.abel533.springboot.Appli
13 cation : Starting Application on liuzh-PC with PID 9844 (F:\Liu\IDEA\SpringBoot\spring-bo
14 ot\target\classes started by liuzh_3nofxnp in F:\Liu\IDEA\SpringBoot\spring-boot)
15 2015-12-12 22:26:35.332 INFO 9844 --- [ main] ationConfigEmbeddedWebApplication
16 Context : Refreshing org.springframework.boot.context.embedded.AnnotationConfigEmbeddedWeb
17 ApplicationContext@a38d7a3: startup date [Sat Dec 12 22:26:35 CST 2015]; root of context h
18 ierarchy
19 2015-12-12 22:26:35.734 INFO 9844 --- [ main] o.s.b.f.s.DefaultListableBeanFact
20 ory : Overriding bean definition for bean 'beanNameViewResolver': replacing [Root bea
21 n: class [null]; scope=; abstract=false; lazyInit=false; autowireMode=3; dependencyCheck=0
22 ; autowireCandidate=true; primary=false; factoryBeanName=org.springframework.boot.autoconf
23 igure.web.ErrorMvcAutoConfiguration$WhitelabelErrorViewConfiguration; factoryMethodName=be
24 anNameViewResolver; initMethodName=null; destroyMethodName=(inferred); defined in class pa
25 th resource [org/springframework/boot/autoconfigure/web/ErrorMvcAutoConfiguration$Whitelab
26 elErrorViewConfiguration.class]] with [Root bean: class [null]; scope=; abstract=false; la
27 zyInit=false; autowireMode=3; dependencyCheck=0; autowireCandidate=true; primary=false; fa
28 ctoryBeanName=org.springframework.boot.autoconfigure.web.WebMvcAutoConfiguration$WebMvcAut
29 oConfigurationAdapter; factoryMethodName=beanNameViewResolver; initMethodName=null; destro
30 yMethodName=(inferred); defined in class path resource [org/springframework/boot/autoconfi
31 gure/web/WebMvcAutoConfiguration$WebMvcAutoConfigurationAdapter.class]]
32 2015-12-12 22:26:36.302 INFO 9844 --- [ main] s.b.c.e.t.TomcatEmbeddedServletCo
33 ntainer : Tomcat initialized with port(s): 8080 (http)
34 2015-12-12 22:26:36.456 INFO 9844 --- [ main] o.apache.catalina.core.StandardSe
35 rvce : Starting service Tomcat
36 2015-12-12 22:26:36.457 INFO 9844 --- [ main] org.apache.catalina.core.Standard
37 Engine : Starting Servlet Engine: Apache Tomcat/8.0.20
38 2015-12-12 22:26:36.537 INFO 9844 --- [ost-startStop-1] o.a.c.c.C.[Tomcat].[localhost].
39 [/] : Initializing Spring embedded WebApplicationContext
40 2015-12-12 22:26:36.537 INFO 9844 --- [ost-startStop-1] o.s.web.context.ContextLoader
41 : Root WebApplicationContext: initialization completed in 1207 ms
42 2015-12-12 22:26:36.941 INFO 9844 --- [ost-startStop-1] o.s.b.c.e.ServletRegistrationBean
43 : Mapping servlet: 'dispatcherServlet' to [/]
44 2015-12-12 22:26:36.944 INFO 9844 --- [ost-startStop-1] o.s.b.c.embedded.FilterRegistrati
45 onBean : Mapping filter: 'characterEncodingFilter' to: [/]
46 2015-12-12 22:26:36.945 INFO 9844 --- [ost-startStop-1] o.s.b.c.embedded.FilterRegistrati
47 onBean : Mapping filter: 'hiddenHttpMethodFilter' to: [/]
48 2015-12-12 22:26:37.111 INFO 9844 --- [ main] s.w.s.m.m.a.RequestMappingHandler
49 Adapter : Looking for @ControllerAdvice: org.springframework.boot.context.embedded.Annotat
50 ionConfigEmbeddedWebApplicationContext@a38d7a3: startup date [Sat Dec 12 22:26:35 CST 2015
51 ]; root of context hierarchy
52 2015-12-12 22:26:37.152 INFO 9844 --- [ main] s.w.s.m.m.a.RequestMappingHandler
53 Mapping : Mapped "{[/],methods=[],params=[],headers=[],consumes=[],produces=[],custom=[]}"
54 onto java.lang.String com.github.abel533.springboot.Application.home()
55 2015-12-12 22:26:37.152 INFO 9844 --- [ main] s.w.s.m.m.a.RequestMappingHandler
56 Mapping : Mapped "{[/now],methods=[],params=[],headers=[],consumes=[],produces=[],custom=
57 []}" onto java.lang.String com.github.abel533.springboot.Application.hehe()
58 2015-12-12 22:26:37.156 INFO 9844 --- [ main] s.w.s.m.m.a.RequestMappingHandler
```



```
Mapping : Mapped "{[/error],methods=[],params=[],headers=[],consumes=[],produces=[],custom
=[]}" onto public org.springframework.http.ResponseEntity<java.util.Map<java.lang.String,
java.lang.Object>> org.springframework.boot.autoconfigure.web.BasicErrorController.error
(javax.servlet.http.HttpServletRequest)
2015-12-12 22:26:37.156 INFO 9844 --- [main] s.w.s.m.a.RequestMappingHandler
Mapping : Mapped "{[/error],methods=[],params=[],headers=[],consumes=[],produces=[text/htm
l],custom=[]}" onto public org.springframework.web.servlet.ModelAndView org.springframewor
k.boot.autoconfigure.web.BasicErrorController.errorHtml(javax.servlet.http.HttpServletRequ
est)
2015-12-12 22:26:37.175 INFO 9844 --- [main] o.s.w.s.handler.SimpleUrlHandlerM
apping : Mapped URL path [/webjars/**] onto handler of type [class org.springframework.we
b.servlet.resource.ResourceHttpRequestHandler]
2015-12-12 22:26:37.175 INFO 9844 --- [main] o.s.w.s.handler.SimpleUrlHandlerM
apping : Mapped URL path [/**] onto handler of type [class org.springframework.web.servle
t.resource.ResourceHttpRequestHandler]
2015-12-12 22:26:37.195 INFO 9844 --- [main] o.s.w.s.handler.SimpleUrlHandlerM
apping : Mapped URL path [/**/favicon.ico] onto handler of type [class org.springframewor
k.web.servlet.resource.ResourceHttpRequestHandler]
2015-12-12 22:26:37.237 INFO 9844 --- [main] o.s.j.e.a.AnnotationMBeanExporter
: Registering beans for JMX exposure on startup
2015-12-12 22:26:37.279 INFO 9844 --- [main] s.b.c.e.t.TomcatEmbeddedServletCo
ntainer : Tomcat started on port(s): 8080 (http)
2015-12-12 22:26:37.280 INFO 9844 --- [main] c.github.abel533.springboot.Appli
cation : Started Application in 2.181 seconds (JVM running for 2.607)
```

最后

以上是Spring Boot基础的内容，有些不全面的地方或者读者有更多疑问，可以查看Spring Boot完整文档(<http://docs.spring.io/spring-boot/docs/1.2.3.RELEASE/reference/html/index.html>)。

关于Spring Boot更多的内容可以继续关注本博客。

版权声明：版权归博主所有，转载请带上本文链接！联系方式：abel533@gmail.com
本文已收录于以下专栏：Spring Boot 入门 (<http://blog.csdn.net/column/details/15358.html>)



wangudongdong (/wangudongdong) 2017-12-04 21:01 26楼

(/wangudongdong) 第一次就写了一个基于Spring Boot 的REST API，用WisdomTool REST client 工具测通，还生成了精美的测试报告和API文档，
<https://github.com/wisdomtool/rest-client>
好资料值得赞！谢谢作者的分享！顶起！

回复

r562253897 (/r562253897) 2017-09-01 15:43 25楼

(/r562253897) Spring boot + Spring Cloud等一系列组件构建庞大的集群架构

回复